BILKENT UNIVERSITY

CS491/492 - Logbook

Berin Su Iyici 22102342

Fall Semester - 2025

August

In August, we started discussing possible senior project ideas and decided to focus on a logistics and warehouse-related problem. I researched how computer vision, real-time monitoring, and optimization could be combined in a warehouse environment. As a group, we identified the main direction of the project as a system that can observe warehouse operations through cameras and help detect anomalies during loading operations. We also started looking for a suitable supervisor whose expertise matched our project idea.

September

In September, we worked on clarifying the project scope and identifying the main problems in warehouse logistics. I focused on understanding possible workflows in a warehouse, especially truck loading, product movement, camera monitoring, and how computer vision outputs could be useful for later decision-making. We also started researching possible innovation experts who could guide us in areas such as computer vision, logistics, real-time streaming, and optimization. I researched possible competition and academic directions in this area, trying to think of ways to adapt these workflows for small and middle scale businesses that might not want to spend much on complex, tech savvy solutions. We also researched and contacted with different innovation experts

October

In October, we applied for the AI mentorship program. We continued shaping the technical direction of the project. I researched different possible computer vision pipelines for the system, including frame-based detection, temporal/event-based detection, model training for scarce data such as anomalies and hybrid rule-based approaches. During this month, we also contacted possible innovation experts and applied to the AI mentorship program. These discussions helped us realize that the system should not only detect objects, but also interpret detections as meaningful warehouse events.

November

In November, we got into the AI mentorship program and was selected by our mentor Onur Bıçakçı. He informed us of current industry practices and affirmed us of our direction for making CV supported systems reachable for SME. He also recommended some academics to consult with and companies to research. After receiving feedback from our mentors, we

refined the high-level architecture of TIRGÖZ. I researched how real-time video streams could be processed using RTSP/WebRTC pipelines and how YOLO-based detection could be connected to backend services to get a feeling of the possible data flow. We discussed backup plans for difficult computer vision cases, including using barcodes or assigning fixed places for items to make product identification more reliable. This month helped us move from a general idea into a more concrete microservice-based system design.

December

In December, we worked on the Analysis and Requirements Report. I contributed to defining system requirements, use cases, and the data flow between the computer vision subsystem, backend, database, and dashboard. I also researched how synthetic data could be used if real warehouse footage or labeled datasets were insufficient. We discussed generating warehouse-like videos and using markers on boxes as a backup or support mechanism for detection. I also worked on database-related use cases such as storing CV events, retrieving camera/ROI configurations, managing products and trucks, and querying event history.

Spring Semester - 2026

January

In January, most of the month was the semester break after the first-semester demo. During this time I reviewed the feedback we received and thought about how the project could be made more realistic for the final implementation. I focused on understanding which parts of the original idea were feasible and which parts needed backup plans. I also started planning possible computer vision approaches for the next semester, especially for detecting warehouse objects, boxes, products, and unsafe situations. After receiving advice from our project supervisor during the final demo session on synthetic data usage. We tried training with synthetic data and mitigating the domain shift with real data. However this approach was not reliable as our dataset was not balanced enough. We decided to steer away to post train on different yolo models for specialiser models and continue with that safer path while experimenting with other approaches.

February

In February, I mainly focused on researching and experimenting with different computer vision models and approaches. Around this time, we had a working yolo specialiser based approach. I tested and compared different possible models for object detection, box detection, damage detection, PPE detection, and warehouse-related object recognition. I

also researched available datasets and tried to understand whether existing models would be reliable enough for our use case.

During this month, we realized that directly identifying products only through general object detection could be unreliable in a real warehouse setting. Because many boxes can look visually similar, we started considering more robust identification methods. I also investigated different deployment formats and model-running options, such as PyTorch, ONNX or OpenVINO, to understand what would be practical for our hardware and Docker-based setup.

Overall, February was mostly a research and experimentation month for my part. Instead of implementing the final pipeline directly, I worked on identifying which computer vision approach could realistically support the MVP.

March

Around this time, we found and seriously considered ArUco and AprilTag markers as a practical solution for product and box identification. This was important because it gave us a more reliable backup plan compared to trying to recognize every product only from appearance.

We worked on understanding how marker detection could be combined with YOLO detections. The idea was that YOLO could detect general objects such as boxes, people, forklifts, and safety equipment, while ArUco/AprilTag markers could help identify specific boxes or products. This helped us define a more realistic computer vision flow for the project.

We also worked on the early version of the MVP pipeline. This included thinking about how detections would be produced, how they would be sent to the backend, and how the frontend or simulation side could use them.

April

I worked on connecting the YOLO service, marker detection logic, backend event handling, and frontend loading workflow. This was the month where the detection system started becoming an actual project pipeline rather than only separate experiments.

I worked on converting raw detections into structured events. These included zone-related events, item movement events, damaged item cases, wrong item scenarios, overload cases, and loading-related events. Zone related events were structured with expandibility in mind for customer requests after the MVP finalised. I also worked on temporal logic, such as detecting when an item enters or exits a loading zone and making sure the same item is not counted multiple times during one pass.

Another major part of April was debugging integration problems. I investigated stream delays, stale bounding boxes, missing events, event payload structure, Docker Compose behavior, backend logs, YOLO worker logs, and frontend visualization issues. I also worked on understanding how loading contexts, trucks, docks, orders, and detected events should fit together.

By the end of April, the project had a much clearer MVP direction. Video input would go through the YOLO/marker pipeline, detections would be converted into events, events would be sent to the backend, and the loading screen would use this information to help monitor warehouse operations.

May

In May, I focused on final integration, debugging, and preparing the system for the final demo and CS Fair. I worked on refining the event and anomaly flow, especially for damaged items, wrong item loading, overload cases, and order progress tracking. I also helped clarify how the loading screen should present information for a truck and dock pair, including loaded quantities, remaining quantities, mismatches, overloads, and unassigned products.

I tested the system using Docker logs, backend endpoints, curl, jq, and frontend behavior. I also worked on making the final explanation of the YOLO path, event generation path, and loading screen path understandable for the presentation. At this stage, the focus was less on researching new models and more on stabilizing the MVP, fixing edge cases, and explaining the project clearly.